| M.A.C.C. NEWLETTER |
| Published Bi-Monthly |

$1.00

From The Vice-President

It was really nice to see so many new faces (or at least ones i haven't seen before) at the last meeting. There are lots of things that we, the executive, would like to accomplish but we need your input and support.

I am no differnt than anyone else. I'm lazy and caught up in my own world with my own problems but for this club to succeed, everyone has to do their best and assist where ever possible. Lets go for it !!!!!!!!

With the theme of the October meeting being HARDWARE MODS and UPGRADES, there are (or should be) a couple of articles in this newsletter that may be of interest. If you have any questions about anything like these projects, please feel free to ask. If the person you do ask doesn't know, we'll try to find an answer somewhere.

Another thing that was brought up at the meeting was a proposal to have a discussion at each meeting about problems any member might be having with hardware, software, or whatever. This type of thing can also be a part of the BBS Message Base, and since some of our newsletters go to the U.S.A. and Australia etc. problems can also be discussed in our newsletter. And you all thought it would be so hard to write an original article.....write about the problems you are having with ????????.

As the guys from Bartles and Jaymes say "THANK YOU FOR YOUR SUPPORT".

Stu Marcellus.

*** ********** ***

Editor's File

From Gary Dikkema

HELLO and WELCOME to the 1987 - 1988 bi-monthly MACC Newsletter!!...I don't think I need too much of an intro...quite a few of you who have modems know who I am (we've probably 'chatted' on 8-Bit) or else chatted at the meetings... soooo without further ado, let's get the show on the road...

Like the past Newsletter Editor I will also enjoy reading all your submitted articles for the Newsletter... please remember to get them to me by the 10th of each second month... This will allow me a little time to "cut and paste" them for the Newsletter...Thanks...

The format of the newsletter will change a bit from past years...It will carry new product reviews (hardware/software)...there will be a 5 part programming tutorial in Action! and Basic for the beginner and advanced programmer (and if there is demand Machine Language may be added in the future)...the "30 second quick game reviews"...interesting (??) articles downloaded from GEnie...hardware/software mods (including diagrams)...club news...and last, but certainly not least contributing articles from club members...

PS A word of caution for those reading the Basic Tutorials, while the examples shows some lowercase instructions remember when you do them to use uppercase.

PSS We were unable to get or print all the files submitted for this edition, so please bear with us...they will appear in the next edition.

*** ********** ***

MINUTES OF 2 SEPTEMBER 1987

PRESENT:

27 CLUB MEMBERS

OPEN MEETING:

STU MARCELLUS OPEN MEETING AT 7:45

MINUTES:

MINUTES OF JUNE 4 READ AND ADOPTED

TREASURER'S:

AS OF:08/11/87

INCOME		EXPENSE	
BALFORWARD:	$569.42		
MEMBERSHIP:	$ 0.00	NEWSLETTER:	$ 40.00
LOTTERIES:	$ 0.00	MAGAZINES:	$ 0.00
INTEREST:	$ 0.00	LIBRARIES:	$ 0.00
ADVERTS:	$ 0.00	OFFICE:	$ 55.00
OTHER:	$ 0.00	MISC.:	$ 0.00
SALES:	4 10.00	BBS:	$ 20.27
CREDITS:	**$579.42**	**DEBITS:**	**$115.96**

BALANCE:$463.46

EXPENSE:$55.69 was for mailing of newsletters. $28.27 was 1 month phone bill and part for BBS

INCOME: $569.42 was carried over from previous executive.$10.00 was proceeds from sale of magazines.

REPORT ADOPTED

LIBRARIAN'S REPORT:

NO REPORT

SYSOP'S REPORT:

THE BBS WILL BE DOWN FOR A FOR FEW DAYS. THE BBS WILL BE CHANGING OVER TO BBS EXPRESS.

NEWSLETTER EDITOR'S REPORT:

RICK SINGBELL HAS RESIGNED AS EDITOR AND GARY DIKKEMA WILL BE THE NEW EDITOR FOR THE NEWSLETTER. GARY WILL PRODUCE A BI-MONTHLY NEWSLETTER STARTING IN OCTOBER. GARY ASK IF ANY WILL DO ANY ARTICLES FOR THE NEWS LETTER.

ATARI FAIR:

RECEIVED A LETTER FROM JAN BERING FROM KILDONAN PLACE STATING THAT DUE TO CONFLICTIONS WITH PERMANENT TENANTS WHO CARRY A WIDE VARIETY OF BRAND NAMES COMPUTERS, WE ARE UNABLE TO ASSIST YOU. PHONE OTHER MALL GOT THE SAME REPLY. ONLY WAY TO DO A MALL DISPLAY IS TO HAVE ALL THE OTHER COMPUTER CLUBS GO IN ON A COMPUTER SHOW AT ONE OF THE MALLS.RAY OF COMPUCENTRE WILL SEE IF THEY CAN HAVE A SIDEWALK SALE AND PUT US IN IT.WILL GET REPORT BACK NEXT MEETING.

AUDITOR'S REPORT:

THAT WE NUMBER ALL MEMBERSHIP CARDS AND CORRESPONDING NUMBERS BE PUT IN A BOUND BOOK SO THAT WE CAN KEEP TRACK OF THE INCOME FROM MEMBERSHIP.

RAY FINCH WILL SUPPLY THE BOUND BOOK FOR THE MEMBERSHIP DIRECTORY.

THEME:

NEXT MEETING THEME WILL BE ON HARDWARE MOD AND UPGRADES.

NEXT MEETING OCTOBER 1,1987 AT GRANT PARK HARROW RECREATION ASSOC.
550 HARROW STREET

7:30 P.M.

MEETING CLOSED 9:30 P.M.

*** ************ ***

MACC: Newsletter
AS OF: 09/18/87

BY: Carole Anderson Treasurer

INCOME:3		EXPENSE:4	
MEMBERSHIP:	$252.98	NEWSLETTER:	$ 76.80
LOTTERIES:	$ 5.00	MAGAZINES:	$ 0.00
INTEREST:	$ 0.00	LIBRARIES:	$ 0.00
ADVERTS:	$ 14.00	OFFICE:	$ 16.65
OTHER:	$ 0.00	MISC.:	$ 0.50
SALES:	$ 0.00	BBS:	$ 23.00
CREDITS:	$271.98	DEBITS:	$116.95

FORWARD: $463.46
BALANCE: $618.49

SUM.EXPENSE: The $16.65 under OFFICE was for the notice in the Free Press. There was a .50 cent service charge from the bank.

SUM.INCOME: We received a $10.00 cheque from the U.S. for a trial membership on which there was an exchange rate of $2.98.

*** ************ ***

Learning programming alone

By: Steve Harnish

Well I've decided to write all these articles to hopefully educate some of the membership. Here is a point by point listing of some of the things to watch out for, rules of the game, and some general guidelines:

-Learning how to program is not an overnight task. At first it will seem like it is impossible, and you will not allow yourself to believe that you could ever write a program of any use or value. Dont let this stage discourage you. Instead take programming one day at a time, and have the attitude that you will eventually get to the end. Believe me there aren't that many things to learn programming wise, its all logic. And if you have any logical abilities at all, you will soon be cranking out tons of code.

-Dont expect anyone to take you by the hand and guide you through it. Sure you will have alot of questions, and you have a little helper in front of you that is just dying for attention...your computer! If you dont know if something will work, try it on the computer, he'll tell you pretty quick if he likes it or not.

-You can never do enough reading! Dont be affraid to look at programming books, or listings of programs in magazines. Just pretend you are the computer and read them line by line. If the listing says 'GOTO 100' then you skip to line 100 and read on.

-Get the memory map for your computer. If you find you are going to program, then you will eventually need this book. It is a book you will never outgrow as it is used constantly. For now you wont need it, but once you finish the beginner section, you cannot go on without it.

-If you get stuck stop. If you are working on a program and it just wont work after a few hours of trying, then stop.

Leave it alone overnight, or for a few days. This allows you time to think about what you are doing, instead of wasting time and getting frustrated in front of the computer.

-Don't use me or anyone else as a crutch! I will know just by the question if you have been trying. I'm not going to be someones book for them. If you ask me something that I know is easy to find in a book, then I'm going to tell you to look it up. By constantly using someone as a crutch you never learn anything.

-Never use equipment as an excuse. I learned programming basic and assembler on my own with an atari 800 and a tape deck. So never come to me with some crying story that you need a printer, and a second drive in order to get anything done!

-People learn at different speeds! Some people will learn programming rather well in a few weeks, others will take years. The important thing is to keep doing it as long as you enjoy it. If your not having fun learning it now, then maybe wait a few months and try it again. The best time to learn programming is in the winter when your sitting home with nothing to do. Since programming envolves your mind you will find it quite easy to go through hours and hours on your computer every night. Ask any programmer if he enjoys programming more than playing games and he will tell you that he does. When you program you accomplish something!

-One final tip, if your ego needs a boost then invite over one of your non-computer friends and show him a few things you have written. They will be impressed, your ego will get a boost, and you will force yourself to work harder so that you can impress them more the next time.

If you run into any problems that you cant solve then use your MACC BBS! There are a whole bunch of members that can help! And for the beginners, dont come to me, your problems should be handled by the intermediate programmers. Lets give me a break and only give me the real toughies!.....

Beginners Basic Lesson 1

by:Steve Harnish

The first thing you have to know in basic is the line format. Each line is numbered, and following the number is the command. For example:

```
10 print "Hello"
```

In this case the line number is 10, and the command is 'print "Hello"'. When you run a basic program it starts at the lowest numbered line and procedes up from there, for example.

```
10 print "Hello"
20 print "Good bye"
```

In this case, 'Hello' is printed, followed by 'Good bye'.

So much for line format, now we move onto the colon. In basic you can cram several statements on one line using the colon. For example the above program could be done line this:

```
10 print "Hello":print "Good bye"
```

There are several advantages of doing this that you will see later, and discover later as well.

If you could only write programs that ran straight from the lowest numbered line to the highest line the computer would not be very useful. Fortunately we have a few tools to make the computer jump over, question, repeat sections of code. This is what makes the computer so powerfull. Let us now look at a few of these tools:

The 'IF' statement:

```
10 X=5
20 If X=5 THEN PRINT "HELLO"
```

In this case, X does equal 5 so the condition is met, and the computer prints 'HELLO'. Look at the IF statement as a door. The word 'IF' asks,'Can you meet this requirement?'. In this case, X=5. Now look at the word 'THEN', and pretend that that is the door. If the requirement is met, the door is opened and the computer does what is behind it.

```
10 X=10
20 If X=5 THEN PRINT "HELLO"
```

In this case the condition is not met so the computer prints nothing.

Now we move onto to the concept of looping. What is looping you ask? Looping is a way to make the computer do something over and over, either forever or for a certain number of times, or till some condition is met. Let us first start with the easiest loop, the GOTO.

```
10 PRINT "HELLO"
20 GOTO 10
```

In this program the computer prints the word 'HELLO' over and over and over forever. This is easy to understand due to the readibility of the 'GOTO' statement. It simply means, every time you get to line 20, GO TO line 10.

Another form of loop is the 'FOR' loop, this is a loop with a condition attached. For example:

```
10 FOR X=1 TO 10
20 PRINT 'HELLO'
30 NEXT X
```

In this program the word 'HELLO' gets printed 10 times. The way it works is as follows:

```
Line 10 X=1
Line 20 PRINT 'HELLO'
Line 30 'Is X=10 yet?'
        If its not, add one to X
        and go to line 10.
```

Luckily the computer handles all that thinking for you, so all you have is a nice neat easy package to work with. The FOR loop is used alot in all programming, and is not limited by just one statement within its 'FOR-NEXT' boundary. For example:

```
10 FOR X=1 TO 10
20 PRINT 'HELLO'
30 PRINT 'ATARI':PRINT 'USERS'
40 NEXT X
```

All the statements between lines 10 and 40 will be executed.

The next type of branching statement is the 'GOSUB' statement. To understand the GOSUB statement you must first understand the concept of a subroutine. A subroutine is a set of lines that are 'jumped' to and then 'returned' from. Rather than try to put this in english, it is easier to understand with an example:

```
10 PRINT 'HELLO'
20 GOSUB 100
30 END
100 PRINT 'GOODBYE'
110 RETURN
```

What happens here is that the computer prints 'HELLO', then it sees,'GOSUB 100'. What that means to it is,'Remember where we are right now, but lets jump over to line 100, and do whatever they want over there till we see a 'RETURN', then well come back here. So the computer runs over to line 100, prints 'GOODBYE', sees the 'RETURN', then comes running back to line 20. It sees theres nothing left to do on line 20, so it goes to line 30. Line 30 tells it that the program is over and the computer stops running.

By mixing the IF, FOR, GOTO, and GOSUB statements together, you have about 80% of your programming knowledge behind you. The rest is just logic, planning, and a knowledge of the finer points of the machine.

*** ************ ***

Intermediate Basic Lesson I
==========

By: Steve Harnish

So you think your intermediate huh? Well I consider an intermediate programmer someone that knows all the basic commands but does not know how to exploit the ataris special features such as scrolling, player missiles etc.etc. So the intermediate lessons will consist each month of how to do 'something' on the atari. The major explanation will be given in the basic lesson followed by the same code in the action lesson. The action lesson however will have other lessons added to it, things that basic cannot do such as pointers, records, etc.etc.

So for this month were going to start with the easiest feature to learn, REDEFINING CHARACTER SETS.

Redefining character sets is a programming aid that is used mostly in games. It can be used in some business software, but mostly it is used in games that require several different screens. I would say that 80% of the games on the atari use this method to handle their animation, game background etc.etc.

A character set is the shape of the letters that the computer dislplays. What this means is that when I type the letter 'A', I get the pattern:

```
  **
 ****
**  **
**  **
******
**  **
```

The atari allows me to change that shape if I want to. It doesnt have to look like an 'A', it can look like a face, a line, an explosion, anything I want!

Each character is 8*8 dots wide. Now that may seem like a small grid to draw shapes into, but dont worry! You can redefine characters so that when they are placed side by side they form the size of the grid you need. For example:

```
AB
CD
```

If I used A,B,C,D close together, I now have a grid 16*16 dots wide. So dont let the 8*8 matrix constrain you. All it takes is carefull planning and a little bit of artistic talent and alot of 'creating'.

Now onto the easy part, the redefinition. There is a three part rule to remember in doing this...
"RESERVE, READ, TELL". Lets look at these three steps seperately.

RESERVE. Your atari has a character set stored in its memory. All this is is a bunch of data that tells it what to make the letters, numbers, and symbols look like. If you RESERVE some memory and put your data in there, you can tell

the atari to look at your data instead of the built in stuff. To reserve memory we need to look at location 106. This location tells the computer how much memory you have for your use. If we look in this location, then subtract some from it, and then put this new number back into 106, the computer 'thinks' there is less memory! You then put your memory up there in the stuff you sneaked from it! For example:

```
TOP=PEEK(106)-5
```

This finds how much free memory there is minus 5 pages. For those of you who dont know, a page is a term used to represent 256 bytes. So now we have TOP which is what we want the computer to think what the top of memory is, so all we have to do to trick it is...

```
POKE 106,TOP
```

There all done you have just saved 5 pages of memory.

READ. In this step we give the computer the new character set data. Sometimes we may have a completely redefined character set, or we may just want to redefine a few characters. If we want to use a complete new set we will read it off of disk and put it in the protected memory. This done with a simple 'FOR' loop.

```
STARTADDRESS=TOP*256
```

Why do we multiply TOP by 256? Well remember that TOP is in pages, and we need to know the address. If this seems confusing just imagine memory as a train. Each car is a page and is equal to 256 bytes. So if we were on the second car we would be at byte number 257 and so on. Now that we have the address its just a simple loop.

```
CLOSE#1:OPEN#1,4,0,"D:DATA"
FOR T=1 TO 1024
GET#1,A
POKE STARTADDRESS+T,A
NEXT T
```

This loop reads in an entire character set. Now if you didnt want to have to redefine an entire set you would first have to move the internal character set to your protected memory. You would do it the same way you do the upper example. All you need to know is that the atari character set starts at 57344. So you just substitute the poke line as follows...

```
POKE STARTADDRESS+T,PEEK(57344 + T)
```

Now this is where being a MACC member has an advantage. What you want to do to save yourself a whole bunch of time is to go get one of the public domain disks that has a character set editor on it. They also have several pre made fonts! So now all you would have to do is modify the ones the club has, and then just save them for your own use!

TELL. This next step is the simplest its the stage where we tell the computer where the character set is. This is quite easy! Memory location 756 is where the computer looks for the first page of its character set. So if we go...

```
POKE 756,TOP
```

The computer will look at our character set!

Now most of you must be thinking that this is all to easy. Well it is! Its an easy job to save memory and tell the computer where the character set is. The hard job is the art work. Redefining the character set to look like you want it to takes hours and hours sometimes. It all depends on what you want, and how much time you have. The best advice I can give you is to start out by modifying a few characters and then taking it from there. Good luck on your artistic abilities and have fun programming!

Here is the complete program that you can type in and check out...

```
10 POKE 106,PEEK(106)-5:GRAPHICS 0
20 STARTLIST=(PEEK(106)+1)*256
30 FOR T=0 TO 1023
32 POKE(STARTLIST+T,PEEK(57344+T)
35 NEXT T
40 POKE 756,STARTLIST/256
```

And thats it! Notice that I did this example a bit differently? Thats because I just want you to know there is more than one way to do things. The differences are subtle, but they are there.....

*** ********** ***

Beginners Action Lesson 1
==========

by:Steve Harnish

The first thing that you will like about Action is that it is similar to basic. For this reason, those reading the Action tutorials should also read the Basic ones. This saves me repeating everything.

The first thing you must realize about action is that the code is structured. What this boils down to is a few extra rules you must follow. The first is that every variable you use must be defined. Now at first this may seem like a drag, but after you get used to it you wont know how you did without it. For example if you have a variable called 'COLLECT' and is defined that way, if you mispelled it,'COLECT' you would get an error. This can save you on hours of debugging, believe me.

Another aspect of action is that the code is broken down into little procedures and functions(more on this later). What this means is that an action program is a whole bunch of little programs laying on top of each other. You will find this extremely powerful later on when you get into bigger programs, but for now dont worry about it.

So lets get started, the first thing we will start with is the header of the procedure. Whats this? For now dont panic, and just blindly follow. Were going to write a little short action program, and just follow along.

```
PROC FIRSTONE()
```

This is our header, and what it means is that we are starting a procedure called 'FIRSTONE' and we are passing no variables to it.

```
BYTE T
```

This is a variable called 'T' and is a byte. A byte has a value from 0-255. There are other data types, but you can consult your manual. Why have all these data types? That is one way to increase speed and help keep our code down to a minimum. If you use an integer(2 byte number) when you should be using a byte, you are using twice the memory, and your making the computer do extra work by juggling two bytes. So if you keep things tight your code will run at prime speed.

```
FOR T=1 TO 10 DO
```

Now here we have a 'FOR' loop in action. As you can see it looks very similar to the basic one. There is one difference, and that is the 'DO'. The 'DO' and 'OD' is how action does loops. The DO starts the loop, and the OD ends it. So for all loops,(FOR,WHILE,UNTIL,DO) we need a DO and an OD.

```
  PRINTE('HELLO')
```

This is a print statement, the 'E' meaning to print an EOL or 'End Of Line' character, so that the cursor goes to the next line.

```
  OD
```

This ends the FOR loop.

```
  RETURN
```

This ends the procedure.

So to sum it up.....

```
  PROC FIRSTONE()

  BYTE T

  FOR T=1 TO 10 DO
    PRINTE('HELLO')
  OD

  RETURN
```

What this proc does is print 'HELLO' ten times.

To keep up with our basic example, lets look at the IF statement. In action the IF statement is much better, as it has 'FI' and 'ELSE'. So for an example.....

```
IF X=5 THEN
  PRINTE('ITS FIVE')
ELSE
  PRINTE('ITS NOT FIVE')
FI
```

What this statement does, is print 'ITS FIVE' if X=5, and it prints,'ITS NOT FIVE' if X is not five. The 'FI' ends the if statement. This is a nice feature as it allows you to have many things happen inside an IF statement.

```
IF X=5 THEN
  PRINTE('HELLO')
  PRINTE('ARE YOU THERE ?')
FI
```

Notice the multiple statements and that I didnt use an 'ELSE'. The else is optional.

There are no 'GOTO' or 'GOSUB' statements in action. The GOSUB is replaced by calling a function or a procedure, but the GOTO just simply does not exist. Why? Well if your code is written correctly there is never a need for a GOTO. At first this may be tough to adapt to, but once you learn it, basic will seem cumbersome.

Intermediate Action Lesson 1

By: Steve Harnish

This lesson parallels the basic one so remember to read the basic lesson and then continue on with this one.

Redefining character sets in action is easier than in basic as you have a few built in routines in your cartridge that are very useful. One of these is MOVEBLOCK. Moveblock is a routine that moves memory from one location to another.

This is useful when moving the built in character set to your reserved memory.

I will first present the code and then we can go through it. This proc can be inserted into your action programs!

```
PROC NEWSET()

BYTE CHBAS=756  ;SEE MAP
BYTE ATARISET=57344  ;START OF SET
CARD LOOP       ;LOOP COUNTER
BYTE RAMTOP=106 ;SEE MAP
BYTE TOP        ;FREE MEMORY
CARD STARTLIST



TOP=RAMTOP-5
RAMTOP=TOP
GRAPHICS(0)
STARTLIST=(TOP-1)*256
MOVEBLOCK(STARTLIST,ATARISET,1024)
CHBAS=TOP-1

RETURN
```

Thats all there is to it! Now all this does is read in the atari set and not any new characters that you have defined. I leave that up to you, it is a simple mater to open a channel to the drive and have a FOR loop read it in.

There are a few things you may want to modify in this program. For example you may want to make TOP a global reference. This is incase you have to do a GRAPHICS call. Remember when you do a graphics call the CHBAS pointer is reset to the original atari character set. This requires you to repeat the last line. You have to retell the computer where your set is. This applies to the basic code as well!

The nice thing about action is once you get this redefined character set proc working you can save it on disk, and then just 'INCLUDE' it in any program of yours in the future! You may wish to make it a function and hav it return the value of TOP. This would allow you to easily re-tell the computer where your character set is.

One thing you may wish to do if you are going to be making alot of graphics calls is write your own mini graphics proc! This sounds hard, but it is painfully simple. Its basically a little trick that would sae you alot of code. Instead of using the GRAPHICS(0) call, you have your own proc called GRAFX. Now the code for this is simple....

```
PROC GRAFX(BYTE MODE)

GRAPHICS(MODE)
CHBAS=TOP-1

RETURN
```

Thats it! See all we have done is used the GRAPHICS proc. Then right after we use the line that tells the computer where to look for the new set. Just remember that CHBAS and TOP would have to be global in this case.

Once you learn how to redefine the character set I suggest you pursue further reading on the subject. There are problems that can arise concerning memory conflicts. These problems will happen to some of you, but you will be few and far between. I dont want to go into details here as I try to keep things as simple as possible for those first attempting these things. Thats the way I learned it, and I know it works. So happy redefining, and use your clubs library! The utilities they have make your job much easier. The alternate character sets they have can be popped into your programs to give them an different feel altogether. Anything from old english flavor to computer lettering. Its

easy to do and is quite impressive. So if you've got the memory for it, use it, dont let it sit there empty.

HOME-MADE 850

(Reprint: EACH,Jan.,1987)

Unlike most computer companies,when Atari designed its 8-bit machines it left out the RS232-C interface to keep down the cost of the computer. To communicate with RS232-C peripherals,such as a modem or printer,requires an 850 interface,which retails for about $200;an expensive accessory. Other interfaces are available with either a serial or parallel port,but usually not both. The "P/R Connection" does offer both,and retails for about $100. It is a cheaper alternative,but requires the loading of a software program to initialize the R:handler,and it may not always be compatible with other software used to run a modem and/or printer. The 850 interface is a computer in itself ,containing a microprocessor and a built in ROM R:handler,and has extensive I/O capabilities. If the cost of an 850 could be reduced to less than $100 it might be the best unit to do the job.

THE $75 850 INTERFACE--Some of the parts needed to build the interface must be ordered from the U.S.,but the rest are available locally. The 850 bare board is available from American TV(check Antic or Analog for ads)for $10 US.You will receive a high quality double-sided Atari factory-original printed circuit board,a crystal(needed to control the microprocessor),and an instruction sheet. Also order two C010750 (6532)PIA chips for $4.50 US each from the same company. The information sheet provides a complete listing of all the parts needed for 4 serial ports and one parallel port. At the end of the sheet is a list of the parts which can be deleted if only one serial port(for a modem)and one parallel port(for a printer)are required,which is the usual case. The 850 ROM chip C0112099B can be ordered from B&C Computervisions for $12 US(check ads as above).

A cheaper alternative is to buy a 2532 EPROM locally for about $10 and burn in the R:handler. The code listing of the ROM is readily available,but you will need access to an EPROM burner. The information sheet details a modification to the board to enable it to use the EPROM chip. All the rest of the parts,including the capacitors,resistors,diodes,transistors, ICs,and IC sockets needed to complete the board are available locally. If you are fortunate enough to have an electronic junk box or scrap circuit boards,you may find you already have many of these items you need. This will reduce the cost of the project even more.

PUTTING THE 850 TOGETHER-- The only tools needed are a set of long-nose pliers to bend and shape the component leads,a small pair of side-cutters to cut off the excess lead from the bottom of the board,a screwdriver to attach the voltage regulator to the heatsink,3 meters of 1.2mm or 0.7mm fine resin core solder,and a 25 watt soldering pencil with a fine tip. If you have a higher wattage soldering pencil,wrapping a copper wire of the desired size around the tip and extending it out to form a new tip will produce good results. Since all the component holes are plated through to both sides,soldering is very easy. Heat the lead and copper pad and apply just enough solder to fill the hole. Another tool you should have is a volt meter. It's a good idea to check the supply voltage values before the chips are plugged in. The parts detailed in the instruction sheet have component numbers (i.e.,C101,R122)which correspond to their location on the circuit board. All the resistors needed,except one,are 1/4 watt size,although 1/2 watt resistors can be used. The signal capacitors,according to the instruction sheet,should be glass or epoxy type. I used the cheaper ceramic disc type capacitors,and they work fine. The size of the signal capacitors is not critical either,as the board is drilled with extra holes to allow up to 3 different sizes. The resistors,crystal, and signal capacitors have no polarity. The electrolytic capacitors are polarized, and the board indicates where the positive lead should go. The signal diodes,rectifiers,and zener diode are also polarized. The band end of the diodes should face the shaded end of the component outline on the board. The LED "power on" indicator should be installed with the flat edge of the base facing the AC power jack. The microchips and the transistors should be socketed to remove the possibility of heat damage,which can occur when soldering them directly to the board. The notch in the microchips should face the notch in the outline of the component on the board. The instruction sheet explains what the proper lead placement is for the transistors. Do not install the microchips into their sockets until after completing assembly of the board,and testing the voltages. One of the voltage regulators(A112)must have a heat sink to dissipate the heat produced by regulating the voltage of the microchips. The heat sink should be about 1-1/2 inches high by 1-1/2 to 2 inches wide,and shaped to form a U to match the outline on the board. Commercial heat sinks are available,but one can be made from any soft metal,like copper. It is not important to solder the heat sink to the board as the original was designed to be. The metal tap of the voltage regulator can be attached directly to the heat sink with a screw and nut.

The selection of I/O plugs,A/C input jack,LED power indicator,printer port,and serial port all depend upon the type of enclosure used to house the board. It is not likely the enclosure will be just the right size to allow the parts to be mounted on the board as the originals were. Select parts which will be easy to mount in your enclosure,and run wires to the board. The parallel printer port can be an RS232-C connector,or any type of connector with at least 13 pins. The

serial modem port can be a 9 pin DB9 type connector.or any 8 pin connector. Make or change your cables to match the connector chosen. The original board had a shield around the microchips,but works fine without one.

POWER-UP AND TEST RUN-- I did change one component from what was called for by the original circuit. The 2 watt resistor(R155)runs across the input to output of the voltage regulator(A112). If the regulator breaks down,more than 5 volts will be applied to the microchips. The 6532 microprocessors have a maximum of 7 volts,and will be damaged by this voltage. To protect the microchips,I left R155 out and put a 5.6 volt,1 amp zener in from ground to the output of A112(the band of the zener faces the output of the regulator). With all the parts in place,except for the microchips,plug in a 9 volt AC power adapter. Set a volt meter to read a positive 10 volts,and connect the ground lead to the heat sink of A112. Connect the positive lead of the volt meter to the non-banded end of CR120. Turn on the power switch and be sure the LED power indicator is on. The volt reader should read 5 volts. Now connect the positive lead to the banded end of CR120. The volt meter should read 10 volts. Change the volt meter to read a negative 10 volts,and connect the positive lead to the emitter of Q102(the lead closest to the AC power jack). The meter should read approximately-8.25 volts. The microchips should not be inserted until these voltages have been obtained. When inserting all the chips,make sure the notch on the end of the chips faces the notch end of the board outline of the chip.

Testing the 850 Interface is very easy. Connect the Serial I/O cable from your computer to the interface and power up the interface. Turn on the computer with the Basic cartridge in,and listen for a multi-tone sound from the speaker. The tone indicates the R:handler has loaded in properly and is acknowledged by the computer. If the interface fails to work properly,check for solder bridges across pads;check all the chips and components to see they are in the correct locations;check to see if all polarities are correct,too. I have had fun building my 850 Interface and have used it with a modem and Hometerm program to communicate with many BBS. The shareware program 850 Express v.3 works great,too. The club library has a copy of the 850 Interface manual,which describes in detail all the functions of the 850,and how to program software to run with it.

Bob Septou

††† ††††††††††† ***

Infiltrator
By Mindscape

As reviewed by Ray Belisle

You are Jonny "Jimbo Baby" McGibbits, ace helicopter pilot, and all around nice guy.

Codename: The Infiltrator...

Infiltrator is Mindscape's newest release for the Atari 8-bit series. The object of the game is that you as 'Jimbo Baby' set out to compleat three missions, each progressively difficult. You must pilot you Gizmo DHX-1 helicopter behind enemy lines, land, compleat your ground mission, and return to home base. (Much more difficult than it sounds!)

While flying, you have total control over your helicopter, but you are not alone in the skies. There are other aircraft, both friendly and unfrielndly, just waiting for you to make a bad move. When you encounter another aircraft you must identify yourself. You do this by sending out the codename of either Infiltrator or Overlord. You must send the correct codename to the other aircraft, eg. Infiltrator to frienly aircraft and Warlord to the enemy. Otherwise you will be met with blaring guns!

If you make it through the flying stage then you must compleat your ground mission. The mission entails that you enter the enemy complex and retrieve the information for which you have been sent. The enemy base is crawling with guards and one wrong move will set off the alarm.

Although the program has great graphics, sound, and some of the most inventive documentation I've seen, the overall game was quite frustrating. When flying, if you happen to give the wrong codename, bend over and kiss your a** goodbye! You're helicopter is supposedly equiped with the latest high-tech equipment available, but I never won one battle in the sky.

If you can make it to the enemy base, it isn't very difficult to land. Although to compleat the ground mission is next to impossible! You have five lives to use but after playing for five hours I don't believe I got close to

finishing. Another annoying feature is that after you have used all five of your lives, you are,transported back to your home base. Essentially you are restarting the game.

The overall game was dissapointing and frustrating. The one bright note is if you know someone who owns the game, borrow the documentation! It has to be some of the best written/most entertaining I've ever read.

###

reprinted from December 86 NYBBLES & BYTES,Combined Atari User Groups,Phoenix,AZ

ATARIWRITER ALERT

Some of us are slightly horrified to discover atariwriter plus operates from a built in version of DOS 2.5. That means it formats a 1050 into the alien "DENSITY AND A HALF",and cannot be used in true double density!. YUK. Well,you can get out of this mess by replacing the DOS on your atariwriter plus disk. Yes,even though it is copyprotected and will not show you a directory(from itself),there is a normal set of DOS(DUP(etc files on the disk. If it is a little scary to do it to your original copy, find someone who can back it up on some modified drive,then write your favorite DOS right to the atariwriter plus disk. Don't worry when it gives you a bad sector error when it tries to write the DUP.SYS file,as it is part of the copyprotection system,and DUP is unused by the atariwriter plus program anyway. I used smartdos,and presto,I have a true double density and auto density changing ATARIWRITER PLUS... with no "1050" density nonsense. This works on both the regular and 130XE versions. (Note: if the DOS you install does not support "AUTORUN.SYS files,be sure to rename the autorun on the ATARIWRITER PLUS disk to whatever name it must be to autoboot under your chosen DOS!)

More atariwriter problems! Although the manual is very specific about the need for every file to end with a RETURN,you probably forget sometimes (I do). What happens next? When you try to print the file,you get a mysterious "FILE LENGTH ERROR" at the bottom of the screen instead of printing. Just go back and add a RETURN to the end of the file and try again.

(Reprinted from the Mid Michigan Atari Magazine)

###

Questions and Answers
==========

By: Steve Harnish

O.K. I admit it these are fake! I made them up! But there is a reason. The main thing I'm doing here is answering most of the problems that people ask over and over. If anyone wants to give me a legitimate question go ahead. Ask away...I dare you.

Q: Whats the difference between LIST and SAVE in basic?

A: When you save a program in basic you save the listing and the variable name table. A LIST saves the program as a text file only. This is handy when you want to print out your code from dos using the C option. But there is a better use of the list command. When you delete a variable in basic, the variable name table remembers it. So if you do a SAVE you are still saving deleted variables! Remember in basic you can only have something like 128 variables. So if you run out, do a LIST, then read it back in with an ENTER and see if that helps you clean things up.

Q: Should I use the other side of my single sided disks?

A: All disks are made in the same way. Then they are tested. The highest grade ones become double sided double density. The next grade is single sided double density. The next and rarer grade is single sided single density. This is not to say that the other side wont work. On the contrary you will find that most people notch the disk and use the other side. I have been doing this for years, and have had no problems. As long as you keep important things backed up do yourself a favor and notch away....

Q: Can I damage my computer with a certain type of program?

A: There is nothing software wise that you can do to damage your computer.

!!! !!!!!!!!!!! !!!

TIP FOR MY DOS USERS

Dear Sir: ....Here's a tip for MYDOS users (v 4.1 to 4.2C)--big upsurge in MYDOS users, as you know, what with it's being shipped with the excellent Newell 256K 800XL upgrade and it's bullet-proof (uncrashable) ramdisk of 1,522 sectors (or 1,010 sectors, protected, with Basic XE!--although I prefer Turbo Basic). Well, to the "Tip" ---> Read all MYDOS extended density sectors from DOS 2.5 with the following two pokes:

```
POKE 4102,234
POKE 4103,234
```

Robert Warren
Englewood, CO

!!! !!!!!!!!!!! !!!

SECRET CODES
(reprint:CLAUG Feb.,1987)

Those of us who use Synfile+ and Syncalc with our 8-bit Atari's may be interested to know we can use both expanded and condensed print with our dot matrix printers.

Hidden on one of the last pages of the Syncalc manual is a short explanation of how to engage condensed print mode for some printers. I'll use Star SG printer codes for examples. To engage condensed print,I press CTRL-O(not zero). The manual doesn't explain how to get back to the standard print mode after engaging condensed print. This started my investigation into the control code caper.

First,I implemented the above example on a test Syncalc spreadsheet. CTRL-O puts a small solid block on the screen(a CTRL-O graphics character). This looks familiar. Researching my files,through old programming books,digging in my closet through old game programs,I finally came to the bottom drawer of my computer cabinet deep below the game paddles and loose printer paper. The answer! A table of ATASCII characters on page 41 of "Understanding Atari Graphics" by Michael Boom shows CTRL-O,decimal 15,produces a solid block. Those of us who own Epson or compatible printers should know decimal 15 engages condensed print. So I went back to the printer manual to see how many other decimal codes are available using this method.

Some printer codes are a single digit. Some others are two digits,and others require multiple commands to enter different print modes. I find a decimal 15 engages condensed print;decimal 18 engages standard pica print. With further research I discovered decimal 14 enters expanded print for one line,and decimal 7 rings the printer's bell. Syncalc does not accept all these different printer codes,so to simplify things,I've included a small table of commands which both Syncalc and Synfile+ accept:

Keystroke	Decimal Code	Function
CTRL-G	7	Ring Bell
CTRL-N	14	Expanded
CTRL-O	15	Condensed
CTRL-R	18	Pica 10 cpi

With Synfile+,you should enter the codes during data entry,not when creating data fields. Combining codes to create expanded condensed print also works,but keep in mind that condensed printing remains engaged even though the expanded print only continues to the end of the line. You must use CTRL-R to cancel condensed mode and return to standard pica print. Good luck in your experimentation!

Editor's (CLAUG) Note: I have tried some of the codes myself on Synfile+ and Syncalc,and have used many more functions

available from the Epson compatible printer I use. I have used underlining ,super-and sub-scripts,double-strike and emphasized. Now some of these codes need the escape key to perform properly. If you are familiar with Synfile+ and Syncalc,you know the escape key is used in these programs to return to menus and the like. I find that to produce the escape character on screen for the printer functions,hold down the SHIFT and press the ESC and you'll get this character! This is not noted in the program manuals.

--Randy Dorn

*** ********** ***

For Sale/Trade Corner

From the Editor

130 XE (expanded to 576K), with 2 - 1050 drives, cassette recorder (410), PR:Connection.... $1,000 Phone Bruce Campbell at 837-9060

*** ********** ***

WATCH THIS SPOT FOR FUTURE ADVERTISEMENTS!!

Manitoba Atari Computer Club

MACC is an independent non-profit user's group with no connections to the Atari Company. We are a group interested in giving the latest news, reviews, rumors and Public Domain software to our members. Material in this newsletter may be copied, as long as credit is given to the newsletter, as well as to the author.

!!! !!!!!!!!!!! ***

This newsletter is produced on an Atari 65XE, 1050 drive (with US Doublers), Atari 850 interface connected to a Panasonic KX-1091 printer using Atari Writer Plus word processing program.

!!! !!!!!!!!!!! ***

MAILING ADDRESS

****

NEW
ADDRESS

****

M.A.C.C.
c/o GARY DIKKEMA
1026 London Street
Winnipeg, Manitoba
R2K 3Y7
Canada

****

NEW
ADDRESS

****

!!! !!!!!!!!!!! ***

Exchange Newsletter
S.L.C.C. Journal
P.O. Box 1506
San Leandro, CA 94577-0374
U.S.A.

M.A.C.C.
c/o Gary Dikkema
1026 London Street
Winnipeg, Manitoba
R2K 3Y7
Canada